**Week 7**

**ReactJS-HOL**

Cricketapp

**Introduction**

This hands-on lab is designed to introduce and reinforce key concepts of ES6 and ReactJS by building a practical application. Through this lab, learners will explore the core features of ES6 such as map(), arrow functions, destructuring, and array merging, while applying them in a real-world React project named **"cricketapp"**. The lab encourages hands-on development of components, logical structuring using conditions, and integration of React with modern JavaScript functionalities.

***Code***

***IndianPlayers.js***

import React from 'react';

const IndianPlayers = () => {

  const T20players = ["Virat", "Rohit", "Pant", "Bumrah"];

  const RanjiPlayers = ["Rahane", "Pujara", "Saha", "Ishant"];

  const allPlayers = [...T20players, ...RanjiPlayers];

  const [first, second, ...rest] = allPlayers;

  const oddTeam = allPlayers.filter((\_, idx) => idx % 2 !== 0);

  const evenTeam = allPlayers.filter((\_, idx) => idx % 2 === 0);

  return (

    <div>

      <h2>Odd Team Players</h2>

      <ul>{oddTeam.map((p, i) => <li key={i}>{p}</li>)}</ul>

      <h2>Even Team Players</h2>

      <ul>{evenTeam.map((p, i) => <li key={i}>{p}</li>)}</ul>

    </div>

  );

};

export default IndianPlayers;

***ListofPlayers.js***

import React from 'react';

const ListofPlayers = () => {

  const players = [

    { name: "Virat", score: 80 },

    { name: "Rohit", score: 50 },

    { name: "Dhoni", score: 90 },

    { name: "Rahul", score: 60 },

    { name: "Pant", score: 85 },

    { name: "Jadeja", score: 45 },

    { name: "Shami", score: 70 },

    { name: "Ashwin", score: 30 },

    { name: "Bumrah", score: 55 },

    { name: "Surya", score: 95 },

    { name: "Gill", score: 77 }

  ];

  const below70 = players.filter(player => player.score < 70);

  return (

    <div>

      <h2>All Players</h2>

      <ul>{players.map(p => <li key={p.name}>{p.name} - {p.score}</li>)}</ul>

      <h2>Players with Score Below 70</h2>

      <ul>{below70.map(p => <li key={p.name}>{p.name} - {p.score}</li>)}</ul>

    </div>

  );

};

export default ListofPlayers;

***App.js***

import React from 'react';

import ListofPlayers from './components/ListofPlayers';

import IndianPlayers from './components/IndianPlayers';

const flag = true;

function App() {

  return (

    <div>

      <h1>Cricket App</h1>

      {flag ? <ListofPlayers /> : <IndianPlayers />}

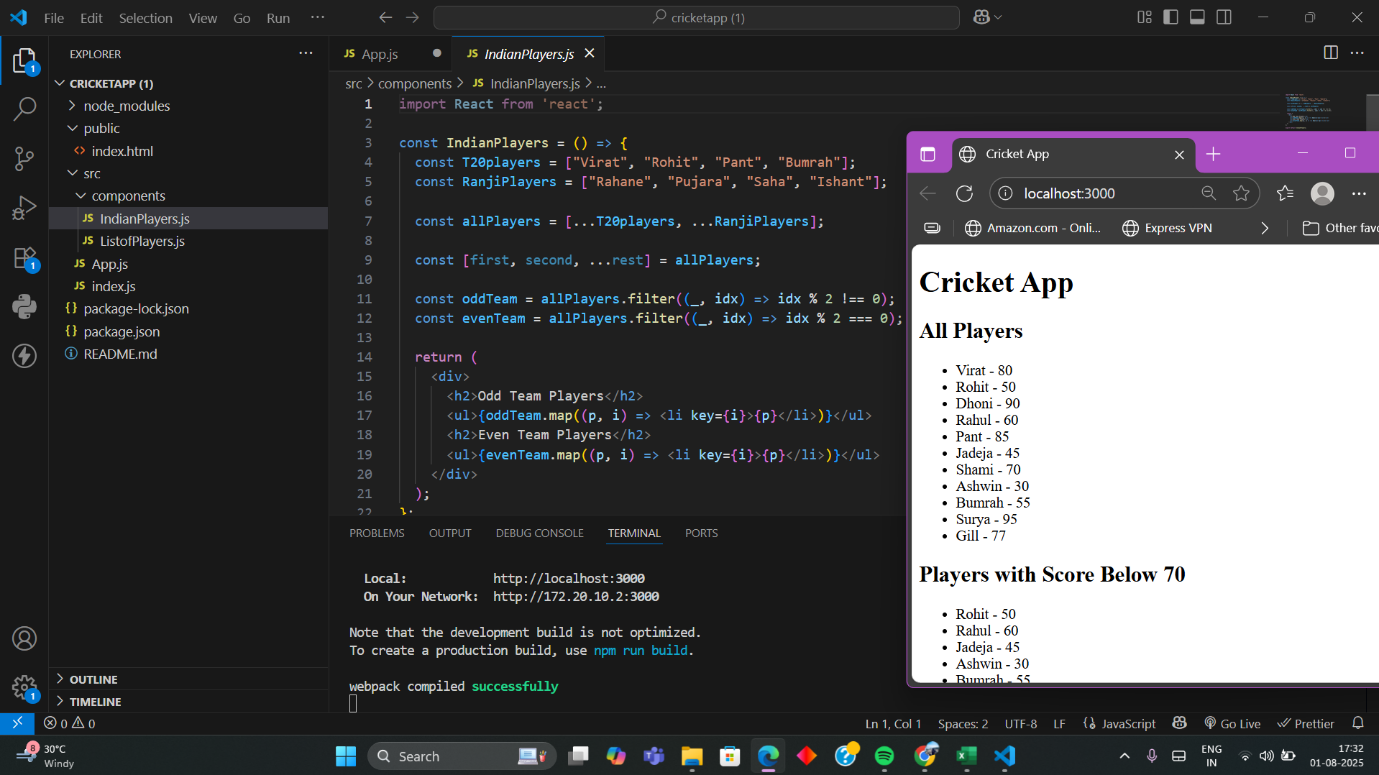
    </div>

  );

}

export default App;

**Screenshot:**



**Conclusion:**

By completing this hands-on lab, learners gain practical experience in implementing modern JavaScript ES6 features within a React application. They become familiar with using map(), arrow functions, destructuring, and array merging techniques, along with the foundational steps of setting up and managing components in React. This exercise not only strengthens their coding skills but also enhances their understanding of React’s component-based architecture and JavaScript’s ES6 capabilities.

**Officespacerentalapp**

**Introduction:**

This hands-on lab is aimed at providing foundational knowledge and practical exposure to JSX in ReactJS. Learners will explore JSX syntax, rendering techniques, usage of JavaScript expressions, and implementation of inline CSS styling. By developing a project titled **"officespacerentalapp"**, learners will apply JSX to dynamically generate UI components and manipulate DOM elements effectively using React conventions

**App.js**

import React from "react";

const App = () => {

  const offices = [

    {

      name: "SkyView Offices",

      rent: 55000,

      address: "MG Road, Bangalore",

      image: "https://via.placeholder.com/200"

    },

    {

      name: "GreenTech Hub",

      rent: 75000,

      address: "Guindy, Chennai",

      image: "https://via.placeholder.com/200"

    },

    {

      name: "SeaView Corporate",

      rent: 62000,

      address: "Marine Drive, Mumbai",

      image: "https://via.placeholder.com/200"

    }

  ];

  return (

    <div style={{ padding: "20px", fontFamily: "Arial" }}>

      <h1>Office Space Rental App</h1>

      {offices.map((office, index) => (

        <div key={index} style={{ border: "1px solid #ccc", padding: "10px", marginBottom: "10px" }}>

          <h2>{office.name}</h2>

          <img src={office.image} alt={office.name} width="200" />

          <p><strong>Address:</strong> {office.address}</p>

          <p style={{ color: office.rent < 60000 ? "red" : "green" }}>

            <strong>Rent:</strong> ₹{office.rent}

          </p>

        </div>

      ))}

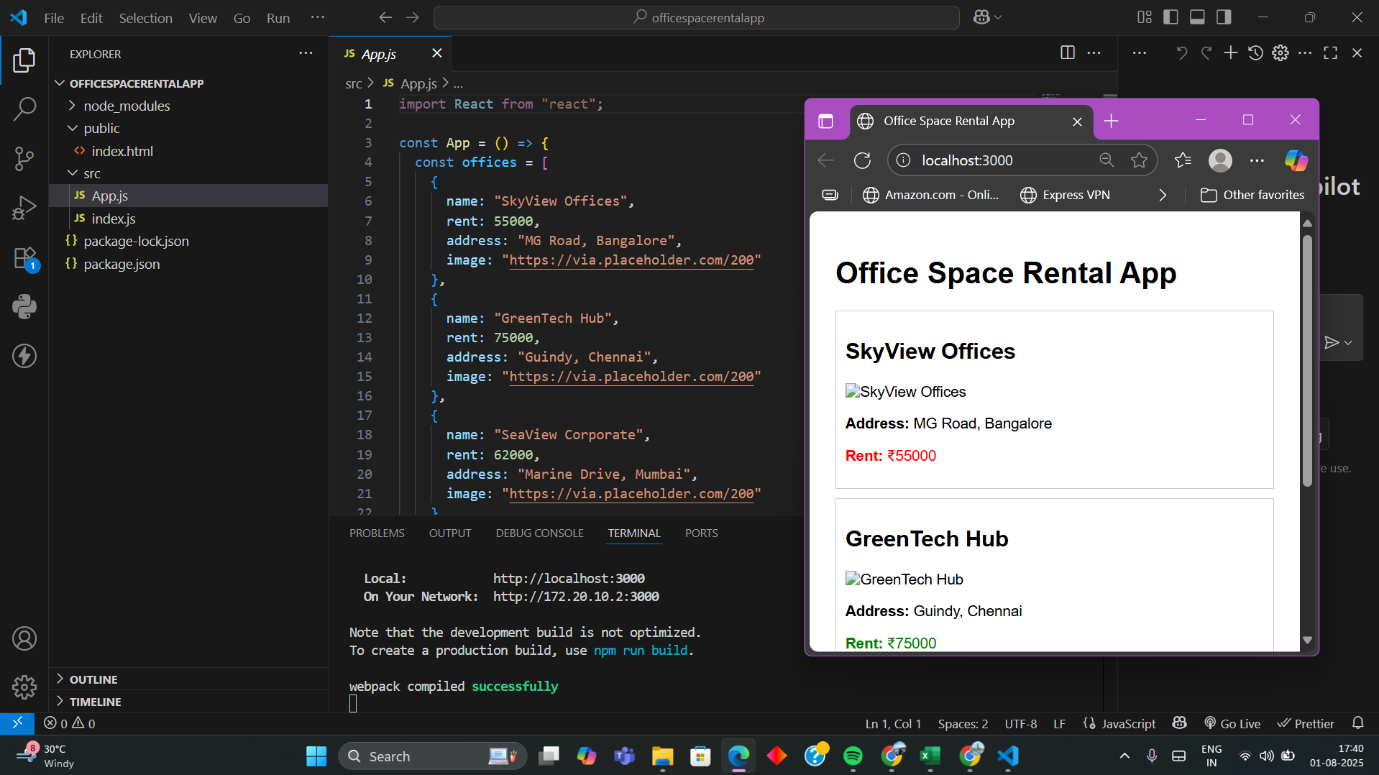
    </div>

  );

};

export default App;

Screenshot:



**Conclusion**

By the end of this lab, learners will have a solid understanding of how JSX enhances React development through its intuitive syntax and seamless DOM integration. They will gain experience in creating elements using JSX, embedding JavaScript logic, and applying conditional inline styling to render dynamic data. This exercise strengthens the bridge between HTML-like syntax and JavaScript logic, a core principle in modern React development.

**EventExamplesApp**

**Introduction**

This hands-on lab introduces learners to the powerful event handling capabilities in React. Through the creation of a project titled **"eventexamplesapp"**, participants will explore key React event concepts including synthetic events, event handlers, and the usage of the this keyword. By working with form elements, buttons, and user-triggered interactions, learners will gain practical experience in managing dynamic behavior within React components.

**Code**

***CurrencyConvertor.js***

import React, { useState } from "react";

const CurrencyConvertor = () => {

  const [rupees, setRupees] = useState("");

  const [euros, setEuros] = useState("");

  const handleSubmit = () => {

    const euroValue = (parseFloat(rupees) / 90).toFixed(2);

    setEuros(euroValue);

  };

  return (

    <div>

      <h2>Currency Convertor</h2>

      <input

        type="number"

        placeholder="Enter amount in INR"

        value={rupees}

        onChange={(e) => setRupees(e.target.value)}

      />

      <button onClick={handleSubmit}>Convert</button>

      {euros && <p>Converted Amount in Euro: €{euros}</p>}

    </div>

  );

};

export default CurrencyConvertor;

***index.js***

import React from "react";

import ReactDOM from "react-dom/client";

import App from "./App";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(<App />);

***App.js***

import React, { useState } from "react";

import CurrencyConvertor from "./CurrencyConvertor";

const App = () => {

  const [count, setCount] = useState(0);

  const sayHello = () => {

    console.log("Hello! This is a static message.");

  };

  const increment = () => {

    setCount(count + 1);

    sayHello();

  };

  const decrement = () => {

    setCount(count - 1);

  };

  const sayWelcome = (msg) => {

    alert("Welcome message: " + msg);

  };

  const handleClick = (e) => {

    alert("I was clicked");

  };

  return (

    <div style={{ padding: "20px", fontFamily: "Arial" }}>

      <h1>React Event Handling Example</h1>

      <p>Counter Value: {count}</p>

      <button onClick={increment}>Increment</button>

      <button onClick={decrement}>Decrement</button>

      <br /><br />

      <button onClick={() => sayWelcome("Welcome!")}>Say Welcome</button>

      <br /><br />

      <button onClick={handleClick}>Synthetic Event (OnPress)</button>

      <br /><br />

      <CurrencyConvertor />

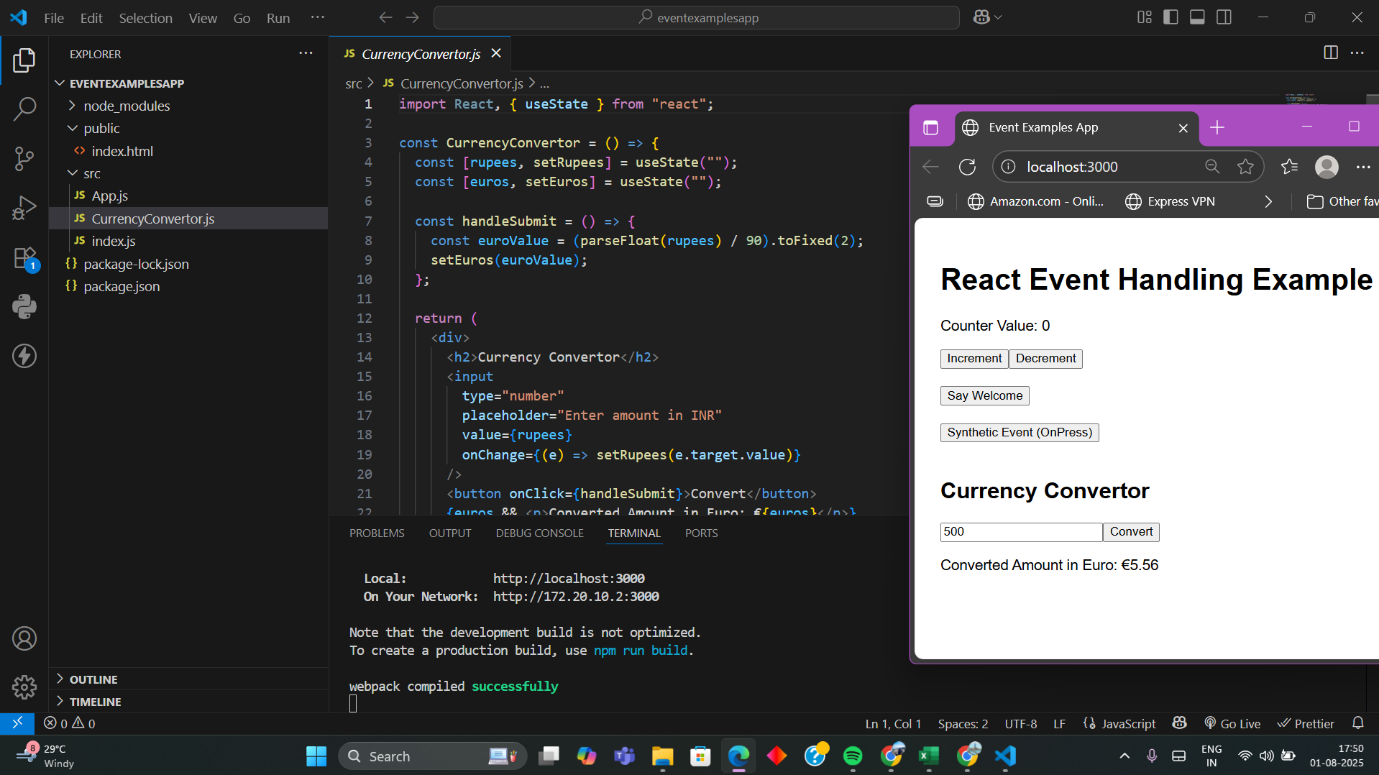
    </div>

  );

};

export default App;

**Screenshot ﻿:**



**Conclusion:**

By completing this lab, learners will have developed a strong understanding of handling events in React applications. They will be able to define and use synthetic events, manage function bindings, and apply logic to user interactions effectively. This exercise reinforces event-driven programming concepts in React and equips learners to build more interactive and responsive user interfaces.

**Ticketbookingapp**

**Introduction**

This hands-on lab focuses on implementing conditional rendering in React applications. By developing a project titled **"ticketbookingapp"**, learners will understand how to dynamically control the display of components based on user interaction or state. The lab highlights the use of element variables and techniques to prevent components from rendering under certain conditions, which are essential for creating user-specific views and secure interfaces.

**Code**

***GuestPage.js***

import React from "react";

const GuestPage = () => {

  return (

    <div>

      <h2>Welcome Guest!</h2>

      <p>Here are some available flights:</p>

      <ul>

        <li>Flight A - Bangalore to Delhi</li>

        <li>Flight B - Mumbai to Chennai</li>

        <li>Flight C - Kolkata to Hyderabad</li>

      </ul>

      <p>Please log in to book your tickets.</p>

    </div>

  );

};

export default GuestPage;

***UserPage.js***

import React from "react";

const UserPage = () => {

  return (

    <div>

      <h2>Welcome User!</h2>

      <p>You can now book your flights.</p>

      <button>Book Flight A</button>

      <button>Book Flight B</button>

      <button>Book Flight C</button>

    </div>

  );

};

export default UserPage;

***App.js***

import React, { useState } from "react";

import GuestPage from "./GuestPage";

import UserPage from "./UserPage";

const App = () => {

  const [isLoggedIn, setIsLoggedIn] = useState(false);

  const handleLogin = () => setIsLoggedIn(true);

  const handleLogout = () => setIsLoggedIn(false);

  return (

    <div style={{ padding: "20px", fontFamily: "Arial" }}>

      <h1>Ticket Booking App</h1>

      {isLoggedIn ? (

        <>

          <UserPage />

          <button onClick={handleLogout}>Logout</button>

        </>

      ) : (

        <>

          <GuestPage />

          <button onClick={handleLogin}>Login</button>

        </>

      )}

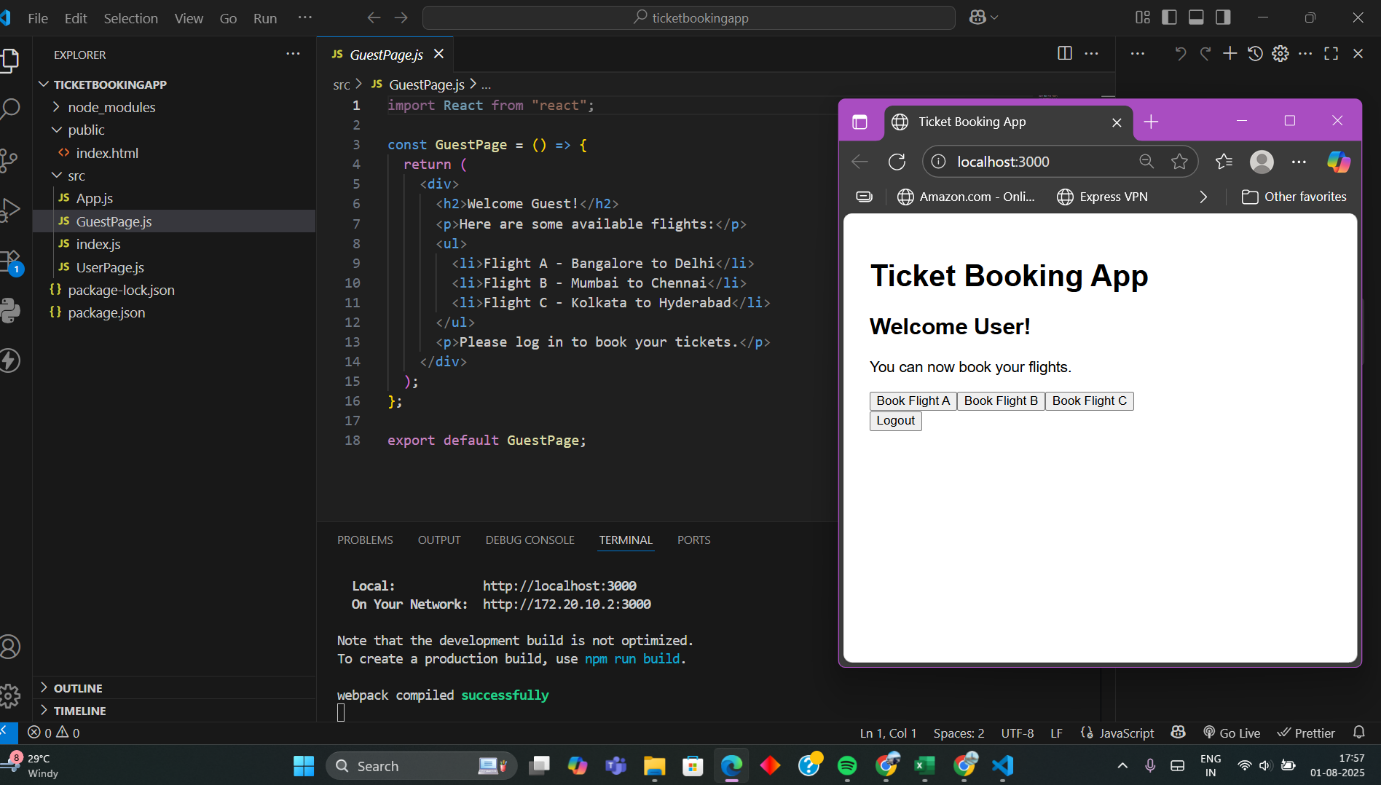
    </div>

  );

};

export default App;

**Screenshot:**



**Conclusion**

Upon completing this lab, learners will have gained practical experience with conditional rendering in React. They will be able to create dynamic user flows such as switching between guest and logged-in views using simple conditional logic. This enhances the user experience by enabling React apps to display content contextually and responsively based on user actions or authentication status.

**Blogger App**

**Introduction**

The objective of this hands-on lab is to explore **conditional rendering** in React by building a simple React application named BloggerApp. This app displays three types of content: **Book Details**, **Blog Details**, and **Course Details**, based on user interaction. Through this project, we understand how to render different components dynamically using React's state and JSX conditional logic.

**Code**

**BlogDetails.js**

import React from 'react';

const blogs = ["Understanding JSX", "State vs Props", "React Lifecycle"];

function BlogDetails() {

  return (

    <div>

      <h2>Blog Details</h2>

      <ul>

        {blogs.map((blog, index) => <li key={index}>{blog}</li>)}

      </ul>

    </div>

  );

}

export default BlogDetails;

**BookDetails.js**

import React from 'react';

const books = ["React Basics", "Advanced React", "React Hooks Guide"];

function BookDetails() {

  return (

    <div>

      <h2>Book Details</h2>

      <ul>

        {books.map((book, index) => <li key={index}>{book}</li>)}

      </ul>

    </div>

  );

}

export default BookDetails;

**CourseDetails.js**

import React from 'react';

const courses = ["ReactJS FSE", "Frontend Mastery", "CTS Training React"];

function CourseDetails() {

  return (

    <div>

      <h2>Course Details</h2>

      <ul>

        {courses.map((course, index) => <li key={index}>{course}</li>)}

      </ul>

    </div>

  );

}

export default CourseDetails;

**App.js**

import React, { useState } from 'react';

import BookDetails from './components/BookDetails';

import BlogDetails from './components/BlogDetails';

import CourseDetails from './components/CourseDetails';

function App() {

  const [view, setView] = useState("book");

  const styles = {

    app: {

      textAlign: "center",

      padding: "30px",

      fontFamily: "'Segoe UI', Tahoma, Geneva, Verdana, sans-serif",

      background: "linear-gradient(to right, #f8f9fa, #e9ecef)",

      minHeight: "100vh",

    },

    heading: {

      fontSize: "2rem",

      marginBottom: "20px",

      color: "#343a40",

    },

    buttonContainer: {

      marginBottom: "30px",

    },

    button: {

      margin: "0 10px",

      padding: "10px 20px",

      fontSize: "16px",

      borderRadius: "8px",

      border: "none",

      backgroundColor: "#007bff",

      color: "white",

      cursor: "pointer",

    },

    buttonHover: {

      backgroundColor: "#0056b3",

    }

  };

  return (

    <div style={styles.app}>

      <h1 style={styles.heading}>Welcome to Blogger App</h1>

      <div style={styles.buttonContainer}>

        <button style={styles.button} onClick={() => setView("book")}>Book Details</button>

        <button style={styles.button} onClick={() => setView("blog")}>Blog Details</button>

        <button style={styles.button} onClick={() => setView("course")}>Course Details</button>

      </div>

      {view === "book" && <BookDetails />}

      {view === "blog" && <BlogDetails />}

      {view === "course" && <CourseDetails />}

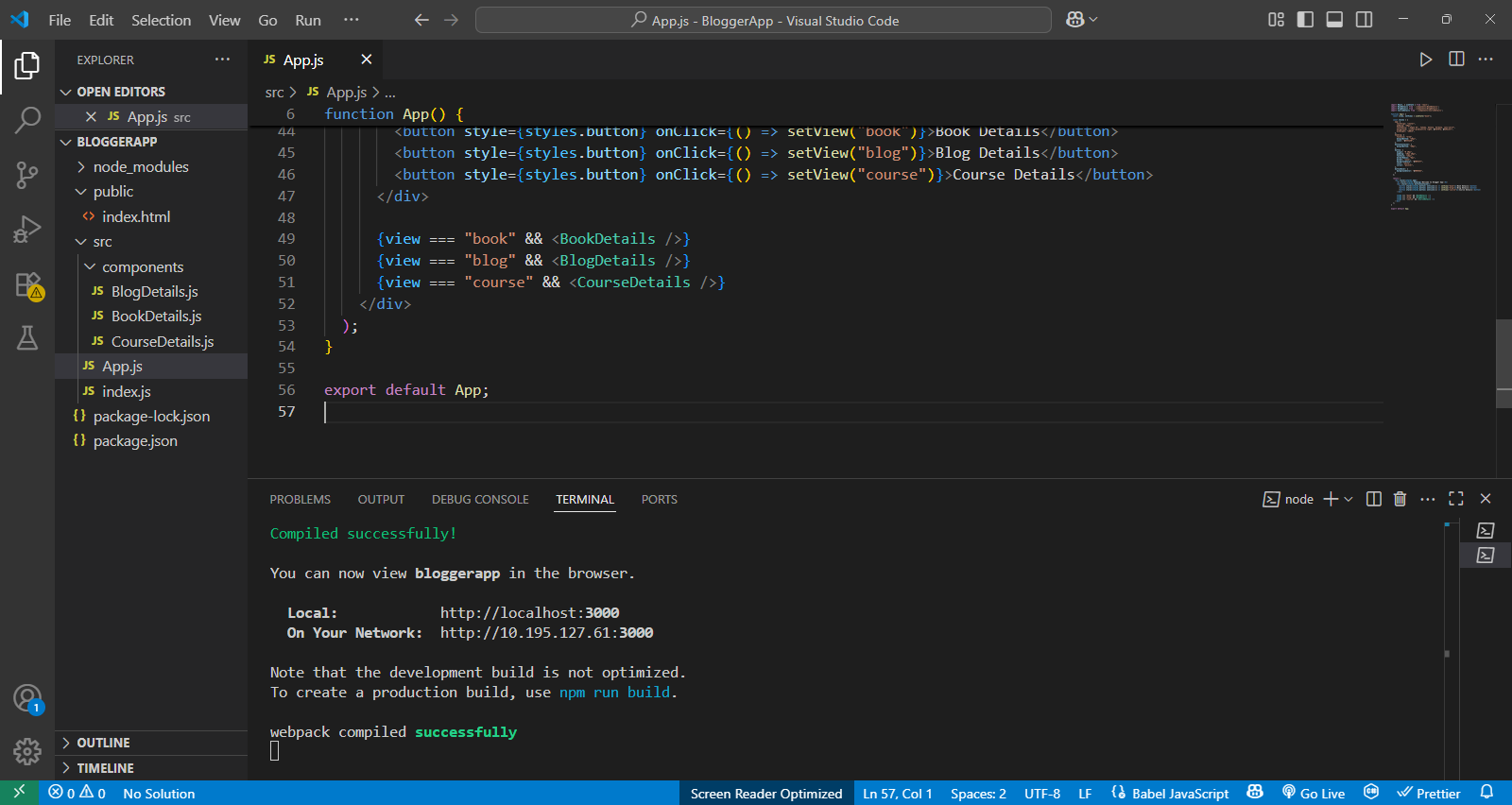
    </div>

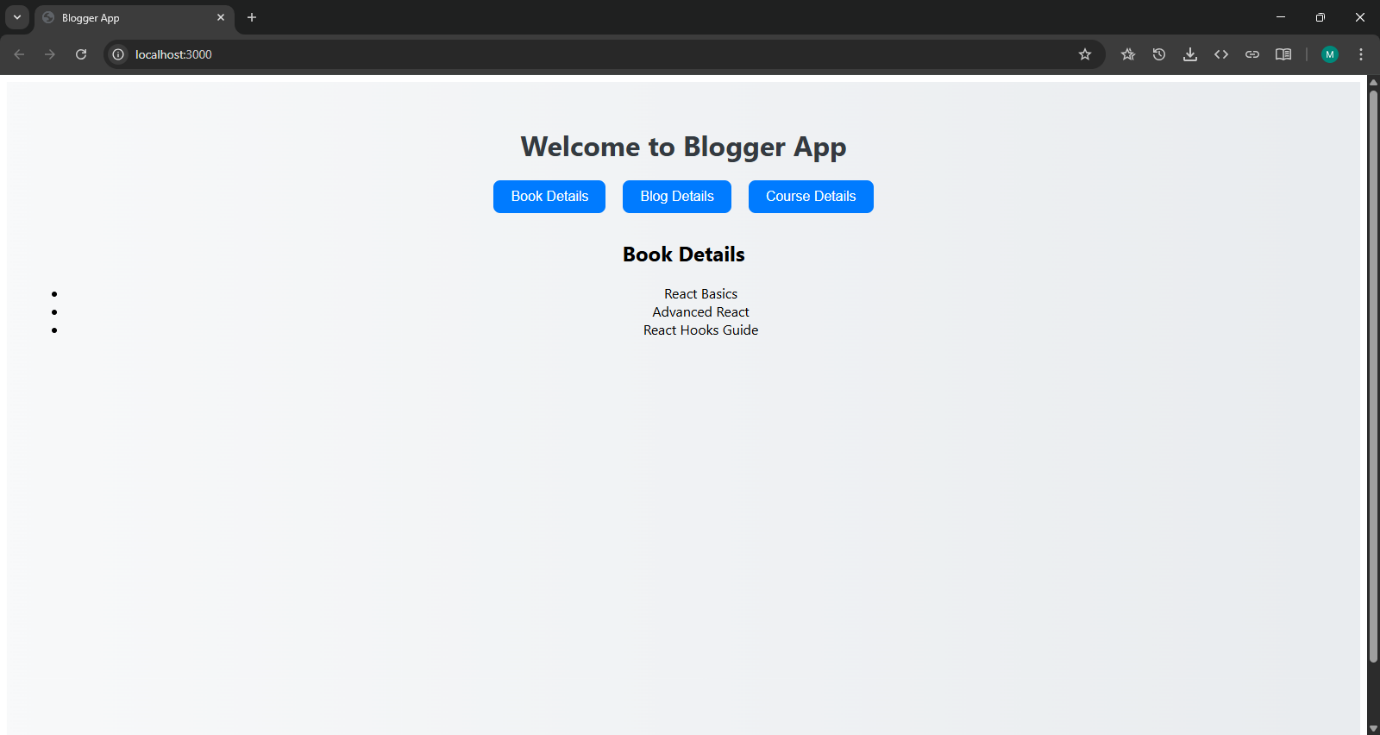
  );

}

export default App;

**Screenshot:**





**Conclusion**

In this hands-on lab, we learned how to use **conditional rendering** in React. By switching between components using useState and conditional logic, we understood key React patterns. This helps in building clean, interactive, and dynamic web apps.